a feedback loop permitting the rasterization stage to return information to the traversal stage; and

a display stage for scanning resulting pixels in frame buffer for display to a display device.

*12. (New) A system comprising:

means for constructing a coarse Z-buffer, the coarse Z-buffer subdivided into a series of tiles, each tile having an associated depth value;

means for updating the depth values of the coarse Z-buffer using Z information from a frame buffer; and

means for using the depth values to selectively discard occluded objects from an image being rendered into the frame buffer.

*13. (New) A machine-readable medium comprising instructions to a machine to:

2-13 canc.

construct a coarse Z-buffer, the coarse Z-buffer subdivided into a series of tiles, each tile having an associated depth value;

update the depth values of the coarse Z-buffer using Z information from a frame buffer; and

use the depth values to selectively discard occluded objects from an image being rendered into the frame buffer.

Remarks

Attached hereto is a marked-up version of the changes made to the specification and claims by the current amendment. The attached page is captioned **"Version with markings to show changes made."** Applicant respectfully requests that a timely Notice of Allowance be issued in this case.

The specification is being amended above to correct minor typographical errors and to improve clarity. No new matter is being added by virtue of the proposed amendment to the specification.

Attached to this Amendment A and Response is a copy of the drawings for Figure 2 with proposed changes shown in red ink. Also attached to this Amendment A and Response is a LETTER TO OFFICIAL DRAFTSPERSON (Request to Make Proposed Drawing Changes) requesting changes to be made in Figure 2. Applicant requests the Examiner's approval for such changes to the drawings for Figure 2. No new matter is being added by the virtue of the proposed amendment to the drawings.

Claims 1-10 were pending the above-identified application when last examined and are amended as indicated above. The claim amendments clarify the claim language and are not intended to limit the scope of the claims, unless the claim language is expressly quoted in the following remarks to distinguish over the art cited.

The Examiner rejected claims 1-10 under 35 U.S.C. 103(a) as unpatentable over Aleksicy (US Patent 5,977, 980) in view of Quaknine et al. (US Patent 6,091,422). Applicant respectfully traverses the rejection.

Aleksicy is directed to a method and an apparatus for determining visibility of a pixel during video rendering by determining z-positioning information of an object element. The z-positioning information is representative of the z-information of an object element in a particular section of the display. A comparison is made between the z-positioning information of the object element and z positioning information stored in temporary memory. When the comparison is favorable, i.e., the object element is in the foreground with respect to the z positioning information stored in the temporary memory and the temporary memory is updated with the z positioning information of the object element. In addition, a z buffer, or z memory is updated with the z information of the object element. Aleksicy divides an object into object elements. For an object with a varying z component, Aleksicy uses the furthest background z information for each object element. "Thus, by generalizing the z parameter for an object element within a particular tile, a quick computation may be performed to determine whether the entire object element, within a tile, is in a foreground with respect to other object elements in the tile. If, however, the quick calculation cannot confirm the object elements being in the foreground or background with respect to other object elements, a complete comparison of the z parameters needs to be performed on a pixel by pixel basis within the tile." (Aleksicy, col.3, line 65 to col.4, line 7). Thus, Aleksicy is concerned with eliminating redundant pixel fragment operations rather than geometry processing.

Quaknine is directed to a computer 3D modeling and animation system providing a user interface that allows an artist to make changes to 3D scene data while simultaneously allowing the author to view rendered images corresponding to those changes. It is an authoring system providing a realistic rendering that may be updated and displayed continuously while simultaneously allowing full access to all authoring tools. The computational burden of the authoring is handled by rendering in an asynchronous parallel thread with respect to that supporting the authoring user interface. The number of pixels that must be ray-traced is reduced by several mechanisms including: limiting the number of objects, limiting the resolution of the display, tailoring the size of the field-of-view window. Also, only portions of the scene that are affected in a visible way by updates are re-rendered. The rendering fits into the authoring context by forming it in a selectable-size region in a wire-frame or other type of view. Also, the authoring user interface thread is executed at a higher priority than the rendering thread to insure the user interface is not slowed down. The rendering is placed on top of a wire frame so that it fits into and takes advantage of the wire-frame view without interfering with it substantially. The rendering window can be tailored to show precisely the part of the scene the author is working on. In Quaknine: "The frame of the rendered image is fragmented into portions. When scene modifications are entered, and a stale rendering is resident in memory, only the portions of the rendering that must be re-rendered are submitted to a rendering task that runs in parallel with the support of the user-interface. In an embodiment of the invention, the subportions of the rendering are regular adjoining shapes referred to as tiles.... The user-interface continually accepts scene modifications and updates abstracted images, such as wire-frame views, in response to the author's work..... Before controlling the rendering process in response to scene modifications, the modifications are checked to determine if they would manifest any visible difference in the render region view.... Constituent portions of the rendered image are sequentially re-rendered as a background task until the portions of the render region required to be updated have been re-rendered.... Storing the rendered tiles in a cache as rendering in completed. Retrieving valid tiles from the cache and refreshing the display by painting the tiles, as they are retrieved, to maintain synchrony between scene updates entered by the user and the displayed rendered image.... Multithreading the above tasks so that the sequence defined by [the various steps] are performed independently (asynchronously)." (Quaknine, col 3.)

In finding claim 1 obvious, the Examiner states: "It is noted that Aleksicy does not explicitly disclose the coarse Z-buffer subdivide [sic] into a series of tiles; however, Alecksicy's computer display being divided into a plurality of tiles suggests the same idea of dividing the Z-buffer into a series of tiles. Furthermore, Quaknine teaches that the step of dividing the Z-buffer into a series of tiles is widely used in the art (Quaknine, col. 9, lines 65-66). Thus, it would have been obvious to one of ordinary skill in the art to combine Quaknine's teaching into Aleksicy's system by culling occluded objects to improve the efficiency of the video graphics."

Independent claim 1 is distinguished over Aleksicy and Quaknine, individually or in combination, by at least by reciting: "A method for culling occluded objects from an image being rendered into a frame buffer; the method, performed by a host processor, comprising: constructing a coarse Z-buffer, the coarse Z-buffer subdivided into a series of tiles, each tile having an associated depth value; updating the depth values of the coarse Z-buffer using Z information from the frame buffer; and using the depth values to selectively discard the occluded objects from the image being rendered." As the Examiner correctly observes, Aleksicy does not disclose "constructing a coarse Z-buffer, the coarse Z-buffer subdivided into a series of tiles". Further, contrary to the Examiner's assertion, col. 9, lines 63-66 of Quaknine state: "The realistic rendering is displayable in a resizable window that can be adjusted to alter a number of pixels contained therein and thereby limit the computational demands of maintaining a realistic rendering." This passage does not teach or suggest dividing the Z-buffer into a series of tiles; nor does it suggest that such practice is widely used in the art. On the contrary, Quaknine discloses the approach of eliminating redundant pixel fragment operations and teaches away from the invention claimed by claim 1.

It would not have been obvious to modify Aleksicy to include Quaknine. As one reference is focused on comparing the z parameters of the object elements (Aleksicy) and the other on the comparison of the z parameters of pixels (Quaknine), there is no suggestion in either reference to combine the two methods that are mutually exclusive.

Further, it would not have been obvious to combine Aleksicy and Quaknine to include these elements because the combination fails to disclose or suggest "constructing a coarse Z-buffer, the coarse Z-buffer subdivided into a series of tiles, each tile having an associated depth value" or "updating the depth values of the coarse Z-buffer using Z information from the frame buffer". Nor would their combination yield the method of claim 1 that compares the z parameters of the tiles: "constructing a coarse Z-buffer, the coarse Z-buffer subdivided into a series of tiles, each tile having an associated depth value; updating the depth values of the coarse Z-buffer using Z information from the frame buffer".

Accordingly, amended claim 1 is patentable over the combination of Aleksicy and Quaknine for at least the foregoing reasons.

The Examiner further suggests that the element of "updating the depth values of the coarse Z-buffer using Z information from the frame buffer" from the amended claim 1 is disclosed by Aleksicy, col 2, lines 43-45 that state: "In addition, a z buffer, or z memory, is updated with the z information of the object element, where the z information equals the z value at each pixel of an object element." Again, Aleksicy is about z values of pixels whereas the amended claim 1 discusses z values of tiles that may or may not correspond to pixels. Aleksicy, therefore, does not disclose or suggest the recited element of the amended claim 1. Accordingly, Applicant submits that amended claim 1 is patentable over Aleksicy and Quaknine, whether considered individually or in combination, for at least this additional reason.

The Examiner further suggests that the element of "using the depth values to selectively discard objects from the image being rendered" from the amended claim 1 is disclosed by Aleksicy, col. 1, lines 36-39 that state: " For the object that is in the background with respect to another object, the pixel information for the portion of the object that is overlapped is discarded." Claim 1 discards entire objects from the image should the algorithm decide that the object is occluded, whereas, Aleksicy's focus again is on pixels and its algorithm discards the pixel information for only those portions of the object that are overlapped. Aleksicy and Quaknine, whether considered individually or in combination, therefore, do not disclose or suggest the recited element of the amended claim 1. Accordingly, Applicant submits that amended claim 1 is patentable over Aleksicy and Quaknine, whether considered individually or in combination, for the additional reasons presented in the foregoing paragraph.

Claims 2-10 depend from or through to to claim 1 and are patentable over Aleksicy and Quaknine, whether considered individually or in combination, for at least the same reasons. Further, claims 2-10 further distinguish over the combination of Aleksicy and Quaknine by reciting additional features.

With respect to claim 4, the Examiner suggests that the element of "constructing a surrogate volume for an object" from the amended claim 4 is taught by Aleksicy, col. 3, lines 25-26 which state: "To render the three images in tile 18, each object element of each object is individually rendered" and further by Quaknine, cols. 18 through 29. Aleksicy teaches individually rendering each object element which teaches away from the tile approach of claim 1, and the surrogate volume approach of amended claim 4. Further, Quakenine columns cited by the Examiner do not include any discussion of a surrogate volume.

The Examiner further suggests that the element of "comparing the nearest Z-values of the surrogate volume to the depth value of a tile that includes the surrogate volume" from the amended claim 4 is taught by Aleksicy, col. 3, line 33 to col. 4, line 7 which state: "For the given tile 18, an initial z positioning information value is assigned and stored in temporary memory. [subsequently a generalized z value is assigned to each object element within that tile.] Thus, by generalizing the z parameter for an object element within a particular tile, a quick computation may be performed to determine whether the entire object element, within a tile, is in a foreground with respect to other object elements in the tile. If, however, the quick calculation cannot confirm the object elements being in the foreground or the background with respect to the other object elements, a complete comparison of the z parameters needs to be performed on a pixel by pixel basis within the tile." The approach of Aleksicy is qualitatively different from the comparison method claimed by amended claim 4. Method of amended claim 4 involves surrogate volumes rather than object elements of Aleksicy; each surrogate volume corresponds to one tile and not more than one tile; and, therefore, only one comparison between the z-values of the surrogate volume and the tile is required per tile. Aleksicy compares the z-values of the object elements within each tile that may and usually does involve more than one comparison; further, if this approach fails in determining the position of the object elements, then a computationally burdensome pixel by pixel comparison becomes necessary. Amended claim 4 is fundamentally less computationally demanding than the method of Aleksicy.

Accordingly, Applicant submits that claim 4 is patentable over Aleksicy and Quaknine, whether considered individually or in combination, for at least these additional reasons.

With respect to claim 6, Aleksicy teaches the z values of the objects within a tile are compared with the z value of the tile, whereas in claim 6, the z values of the tiles that are spanned by a surrogate volume are compared with the z value of the surrogate volume. Accordingly, claim 6 is selected to be allowable for at least this additional reason.

With respect to claim 8, the Examiner suggests that "constructing a lower resolution coarse Z-buffer, the lower resolution coarse Z-buffer subdivided into series of tiles, each tile having an associated depth value; and updating the depth values of the lower resolution coarse Z-buffer using Z information from the frame buffer" is taught by Quaknine, col. 8, lines 48-50 stating: "Another alternative to limiting the computational demands of the rendering is to allow the user to adjust the resolution of the realistic rendering."

However, contrary to the Examiner's assertion, "adjusting the resolution of realistic rendering," as disclosed by Quaknine, does not teach constructing a lower resolution coarse Z-buffer subdivided into series of tiles or updating the depth values. Accordingly, Applicant submits that claim 8 is distinguishable over Aleksicy and Quaknine, whether considered individually or in combination, for this additional reason.

With respect to claim 9, the Examiner suggests that "each tile in the lower resolution coarse Z-buffer covers the same screen area as each tile in the coarse Z-buffer" is taught by Quaknine, col. 8, lines 48-64 stating: "Another alternative to limiting the computational demands of the rendering is to allow the user to adjust the resolution of the realistic rendering. In a variation, the realistic rendering is subdivided into portions, selected portions being rendered sequentially by the second synchronous thread. The selected portions are those that are determined to change appearance as a result of the update data. This also reduces the computational demand by not reproducing rendered portions. In another variation, the second thread is interrupted when the update data is received and reinstated after the update data becomes available to the rendering process of the second thread. This insures that data, invalidated by update data, is not used in the rendering process. Invalidated portions may be blanked in a display prior to re-rendering so stale parts of the rendering are never displayed. Alternatively, the stale portions remain the display until overwritten by the updated portions."

However, the above passage of Quaknine does not teach or suggest that "each tile in the lower resolution coarse Z-buffer covers the same screen area as each tile in the coarse Z-buffer"; there is no mention of tile system; no mention of various resolutions of the coarse Z-buffer; and no mention that the tiles in both the coarse Z-buffer and lower resolution versions of the coarse Z-buffer cover the same area. The Quaknine passage, rather, speaks of subdividing the realistic rendering into portions and selected portions being rendered. The computational demand, in Quaknine, is reduced by not reproducing rendered portion or the data invalidated by the update data as compared against claim 9 wherein reducing the computational demand is accomplished by using Z-buffers of various resolutions. Accordingly, Applicant submits that claim 9 is distinguishable over Aleksicy and Quaknine, whether considered individually or in combination.

For the foregoing reasons, Applicant requests reconsideration and withdrawal of this rejection under 35 USC 103.

In summary, claims 1-10 were rejected. This response amends claims 1-10 and adds new independent claims 11, 12, and 13. New independent claims 11, 12, and 13 recite features that are not disclosed or suggested by the cited references, whether the cited references are considered singly or in combination. For the foregoing reasons, Applicant respectfully requests allowance of claims 1-13.

If the undersigned attorney has overlooked a teaching in any of the cited references, that is relevant to the allowability of the claims, the Examiner is requested to specifically point out where such teachings may be found.

If for any reason an insufficient fee has been paid, the Examiner is hereby authorized to charge the insufficiency to Deposit Account No. 05-0150.

If the Examiner has any questions or needs any additional information, the Examiner is invited to telephone the undersigned attorney at (650) 843-3355.

Respectfully submitted,
Angus Dorbie

By Fariba Sirjani
Fariba Sirjani
Attorney for Applicant
Reg. No. 47,947

Dated: May 4, 2001
Squire, Sanders & Dempsey L.L.P.
600 Hansen Way
Palo Alto, CA 94304-1043
Telephone (650) 856-6500
Facsimile (650) 856-3619

CERTIFICATE OF MAILING

I hereby certify that this paper (along with any paper referred to as being attached or enclosed) is being deposited with the United States Postal Service on the date shown below with sufficient postage as first class mail in an envelope addressed to the Commissioner for Patents, Washington, D.C. 20231, on

Date 5/4/01 By: [illegible]
Linda Clinkenbeard

Attachment: Version With Markings To Show Changes Made

Version With Markings To Show Changes Made

In the Specification:

Paragraph beginning at page 1, line 7, is being amended as follows:

-- Computer systems (and related devices) typically create three-dimensional images using a sequence of stages known as a graphics pipeline. During early pipeline stages, images are modeled using a mosaic-like approach where each object is composed of a collection of individual points, lines and polygons. These points, lines and polygons are ~~know~~ known as primitives and a single image may require thousands, or even millions, of primitives. Each primitive is defined in terms of its shape and location as well as other attributes, such as color and texture. --

Paragraph beginning at page 1, line 15, is being amended as follows:

-- The graphics pipeline maps, or renders, each primitive into a memory storage device ~~know~~ known as a frame buffer. Each storage location within the frame buffer defines one pixel within the image being produced. The graphics pipeline performs the rendering process by determining which pixels (i.e., which frame buffer storage locations) are included within each primitive. Each pixel is then ~~initializes~~ initialized to reflect the attributes of the primitive, or primitives in which it is included. In many cases, the graphics pipeline will further modify the pixel values in the frame buffer to apply texture, lighting and other effects to the graphics primitives. --

Paragraph beginning at page 3, line 7, is being amended as follows:

-- The present invention provides a method and apparatus for early occlusion culling. For the present invention, the screen is divided into a series of tiles arranged as a rectangular grid. The rectangular grid is known as a coarse Z-buffer and may have various sizes and dimensions. For the purposes of this description, a size of two-hundred and fifty-six tiles arranged in a sixteen by sixteen grid may be assumed. Each tile within the coarse Z-buffer has an associated depth value. Each tile's depth value is defined as the ~~farther~~ farthest Z-buffer value that is included within that tile. --

Paragraph beginning at page 4, line 3, is being amended as follows:

-- The depth values are stored in a location ~~that~~, such as main memory, where they are available to application programs. This allows application programs to reference these values while they are creating graphics images. The program rendering an image constructs a surrogate volume for each object that it adds to the image. The program then compares the nearest Z-value of the surrogate volume to the depth value of the tile that includes the surrogate volume. Based on this comparison, the application program determines if the object is occluded and can be discarded. --

Paragraph beginning at page 5, line 5, is being amended as follows:

-- Advantages of the invention will be set forth, in part, in the description that follows and, in part, will be understood by those skilled in the art from the description herein. The advantages of the invention will be realized and attained by means of the elements and combinations particularly pointed out in the appended claims and <u>their</u> equivalents. --

Paragraph beginning at page 7, line 7, is being amended as follows:

-- In Figure 1, a computer system 100 is shown as a representative environment for the present invention. Structurally, computer system 100 includes a host processor~~, or host processor 102,~~ <u>102</u> and a memory 104. An input device 106 and an output device 108 are connected to host processor 102 and memory 104. Input device 106 and output device 108 are connected to host processor 102 and memory 104. Input device 106 and output device 108 represent a wide range of varying !/O devices such as disk drives, keyboards, modems, network adapters, printers and displays. Each node 102 may also ~~includes~~ <u>include</u> a disk drive 110 of any suitable ~~disk drive~~ type (equivalently, disk drive 110 may be any non-volatile mass storage system such as "flash" memory). Computer system 100 also preferably includes a graphics processor 112 of any suitable type. Graphics processor 112 implements all of the tasks required to translate graphics primitives and attributes to displayable output. In Figure 1, host processor 102 and graphics processor 112 are interconnected using a bus. It should be appreciated that the present invention is equally suitable to environments where host processor 102 and graphics processor 112 share a commonly addressable memory. --

Paragraph beginning at page 8, line 1, is being amended as follows:

-- Computer system 100 is the hose for a graphics pipeline. An implementation for this pipeline is designated 200 in Figure 2. Pipeline 200 includes generation stage 202, traversal stage 204, transformation stage 206, rasterization stage 208 and display stage 210. Generation stage 202 corresponds to the creation, acquisition, or modification of information to be displayed and organizing this information into application data structures. Traversal stage 204 corresponds to the ~~of~~ traversal of the application data structures generated in the preceding stage, passing on the appropriate graphics data. Transformation stage 206 corresponds to the transformation of the graphics data from object-space coordinates into eye-space coordinates, performing requested lighting operations, ~~then~~ clipping the transformed data in clip-space~~-~~, and projecting the resulting coordinates into window space. Rasterization stage 208 renders window space primitives (~~like~~ such as points, lines, and polygons) into a frame buffer. Per-vertex shading calculations, texture lookups and calculations, and per-pixel operations ~~like~~ such as depth testing are performed in this stage. Display stage 210 scans the resulting pixels in the frame buffer, typically for display to a video monitor or other device. --

Paragraph beginning at page 9, line 7, is being amended as follows:

--The present invention provides a method and apparatus for early occlusion culling. The method and apparatus of the present invention are better understood by reference to representative image 300 of Figure ~~2~~ 3. Image 300 depicts a runway partially obscured by clouds. ~~Image 300 is shown with an eye point 202.~~ Eye-point ~~202~~ 302 represents the position at which image 300 is viewed. Eye-point ~~202~~ 302 is intended to be movable in relation to image 300. This means that components of image 300, including the runway and clouds may be viewed from a range of positions. --

Paragraph beginning at page 10, line 3, is being amended as follows:

-- Graphics pipeline 200 is configured to update the depth values in memory 104. ~~For the particular configuration shown, this~~ This means that, for the particular configuration shown, rasterization stage 208 uses feedback loop 212 to continuously update the depth values within memory 104. ~~It should be noted that different~~ Different configurations may perform this update as either a "push" or "pull" operation. Thus, for some configurations, rasterization stage 208 will transfer (push) depth values to memory 104. In other configurations, another entity (such as traversal stage 204 or the graphics application program executing on host processor 102) will retrieve (pull) depth values from rasterization stage 208. The depth values may also be updated either synchronously or asynchronously. For synchronous updating, depth values are transferred to memory 104 as they change within rasterization stage 208. Typically this means that depth values are updated each time corresponding Z-values in a Z-buffer used by rasterization stage 208 are changed. Graphics pipeline 200 may also be configured to perform these updates on a less frequent, asynchronous basis. In many cases this means that graphics pipeline 200 will perform these updates on a periodic basis. In other cases, graphics pipeline 200 can be configured to perform updates whenever a predefined number of changes have taken place in the Z-buffer used by rasterization stage 208. Asynchronous updating provides a balanced approach that retains most of the ~~benefit~~ benefits of the occlusion culling method while reducing the amount of data that must be fed back from the Z-buffer. --

Paragraph beginning at page 13, line 1, is being amended as follows:

-- If the nearest Z-value of the object is less ~~that~~ than the depth value, the object may be visible and method 400 continues at step 416. In step 416, host processor 102 performs the steps required to render the object being processed. Typically, this means that host processor takes whatever steps are ~~require~~ required to pass the object being processed to graphics processor 112. In other cases, host processor 102 may perform the rendering process itself. --

Paragraph beginning at page 14, line 3, is being amended as follows:

-- Each coarse Z-buffer 304 in a series spans the same image 300 at a different resolution. The highest resolution coarse Z-buffer 304 splits image 300 between tiles 306 with no overlap. This is the case shown for coarse Z-buffer 304 of Figure 3. For lower resolution coarse Z-buffers 304, each tile 306 covers a larger area of image 300. Since the total number of ~~tile~~ tiles 306 remains constant, this means that tiles 306 within lower resolution Z-buffers 304 overlap each other. In fact, each succeeding lower resolution Z-buffer 304 includes successively larger tiles 306 that have successively larger overlapping areas. --

In the Claims:

All of the claims are being amended in this response and three new claims are added. Claims that are being amended in this response or are being added are identified with an asterisk.

*1. (Once Amended) A method for culling occluded objects from an image being rendered into a frame buffer; the method, performed by a host processor, comprising ~~the steps, perfumed by a host processor of~~:

constructing a coarse Z-buffer, the coarse Z-buffer subdivided into a series of tiles, each tile having an associated depth value;

updating the depth values of the coarse Z-buffer using Z information from the frame buffer; and

using the depth values to selectively discard the occluded objects from the image being rendered.

*2. (Once Amended) A method as recited in claim 1, ~~where in~~ wherein ~~the step of~~ updating the depth values is performed synchronously as information in the frame buffer changes.

*3. (Once Amended) A method as recited in claim 1, ~~where in~~ wherein ~~the step of~~ updating the depth values is performed asynchronously.

*4. (Once Amended) A method as recited in claim 1, wherein ~~the step of~~ using the depth values to selectively discard the occluded objects further comprises ~~the step of~~:

constructing a surrogate volume for an object; and

comparing ~~the~~ nearest Z-value of the surrogate volume to the depth value of a tile that includes the surrogate volume.

*5. (Once Amended) A method as recited in claim 4, further comprising ~~the step of~~ transforming the surrogate volume from object space to eye space.

*6. (Once Amended) A method as recited in claim 1, wherein ~~the step of~~ using the depth values to selectively discard the occluded objects further comprises ~~the steps of~~:

constructing a surrogate volume for an object; and

retrieving ~~the~~ greatest depth value from the depth values of set of tiles that are spanned by the surrogate volume; and

comparing the nearest Z-value of the surrogate volume to the retrieved depth value.

*7. (Once Amended) A method as recited in claim 6, further comprising ~~the step of~~ transforming the surrogate volume from object space to eye space.

*8. (Once Amended) A method as recited in claim 1, further comprising ~~the step of~~:

constructing a lower resolution coarse Z-buffer, the lower resolution coarse Z-buffer subdivided into a series of tiles, each tile having an associated depth value; and

updating the depth values of the lower resolution coarse Z-buffer using Z information from the frame buffer.

*9. (Once Amended) A method as recited in claim 8, wherein each tile in the lower resolution coarse Z-buffer covers the same screen area as each tile in the coarse Z-buffer.

*10. (Once Amended) A method as described in claim ~~9~~ 8, wherein the tiles in the lower resolution coarse Z-buffer are overlapping.

*11. (New) A system, used as a host for a graphics pipeline, comprising:

a host processor executing a graphics application program, wherein the graphics application program is capable to implement:

a generation stage for creation, acquisition, and modification of information to be displayed, and organizing the information into application data structures; and

a traversal stage for traversal of the application data structures, and passing on appropriate graphics data; and a graphics processor, communicatively couples to the host processor, capable to implement:

a transformation stage for transformation of graphics data from object-space coordinates into eye-space coordinates, performing requested lighting operations, clipping the transformed data in clip-space, and projecting resulting coordinates into window-space;

a rasterization stage for rendering window-space primitives into a frame buffer, and performing shading calculations, texture lookups and calculations, and per-pixel operations;

a feedback loop permitting the rasterization stage to return information to the traversal stage; and

a display stage for scanning resulting pixels in frame buffer for display to a display device.

*12. (New) A system comprising:

means for constructing a coarse Z-buffer, the coarse Z-buffer subdivided into a series of tiles, each tile having an associated depth value;

means for updating the depth values of the coarse Z-buffer using Z information from a frame buffer; and

means for using the depth values to selectively discard occluded objects from an image being rendered into the frame buffer.

*13. (New) A machine-readable medium comprising instructions to a machine to:

construct a coarse Z-buffer, the coarse Z-buffer subdivided into a series of tiles, each tile having an associated depth value;

update the depth values of the coarse Z-buffer using Z information from a frame buffer; and use the depth values to selectively discard occluded objects from an image being rendered into the frame buffer.

In the Drawings:

Applicant seeks permission to change Figure 2. In Figure 2, the feedback loop 212 is mis-titled as 208 while 208 corresponds to the rasterization stage which is correctly marked 208. The corrected Figure 2, changes the number corresponding to the feedback loop from 208 to 212. A red-lined and corrected Figure 2 is enclosed with this attachment.